BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

A2

(12) UK Patent Application (19) GB (11) 2 383 854 (13) A

(43) Date of A Publication 09.07.2003

(21) Application No 0121586.2

(22) Date of Filing 06.09.2001

(71) Applicant(s)
Sun Microsystems Incorporated
(Incorporated in USA - Delaware)
M/S UPAL01-521, 901 San Antonio Road,
Palo Alto, CA 94303,
United States of America

(72) Inventor(s)
Ryan Fintan
John L Ward

(74) Agent and/or Address for Service
Haseltine Lake & Co
Imperial House, 15-19 Kingsway,
LONDON, WC2B 6UD, United Kingdom

(51) INT CL[7]
G06F 11/22

(52) UK CL (Edition V)
G4A AFLB

(56) Documents Cited
GB 2353373 A **GB 2203869 A**
US 6223272 B1 **US 5963743 A**

(58) Field of Search
UK CL (Edition V) **G4A**
INT CL[7] **G06F**
Other: **Online; EPODOC, JAPIO, WPI.**

(54) Abstract Title
Method for checking a computer system's configuration against it's required specification using checksums files.

(57) The configuration of a computer system is checked by providing a file with the required configuration or specification, deriving a second check file by testing the computer's configuration and comparing to two files to approve the computer. The check file could include details of the system's hardware, firmware and software. Also the check files could comprise checksum data for the software components. Also disclosed are a computer program, method of manufacturing a computer and a method of checking a computer after installation.

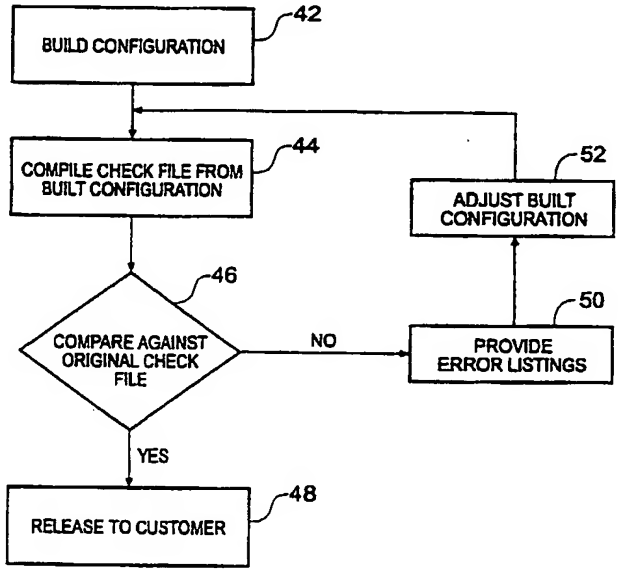

Fig. 5

GB 2 383 854 A

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

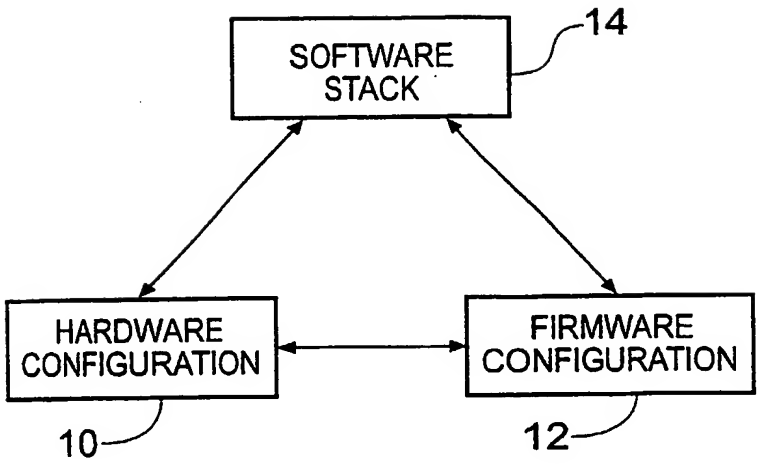
SOFTWARE STACK
14
HARDWARE CONFIGURATION
10
FIRMWARE CONFIGURATION
12

Fig. 1

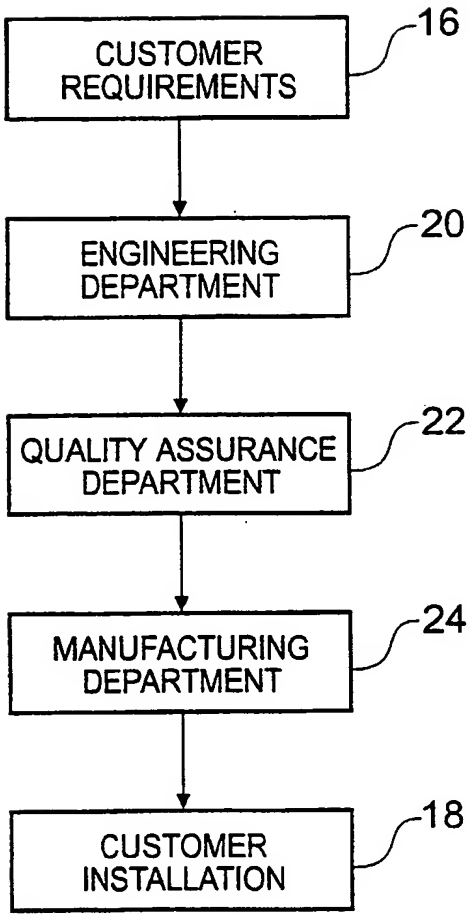
CUSTOMER REQUIREMENTS
16
ENGINEERING DEPARTMENT
20
QUALITY ASSURANCE DEPARTMENT
22
MANUFACTURING DEPARTMENT
24
CUSTOMER INSTALLATION
18

Fig. 2

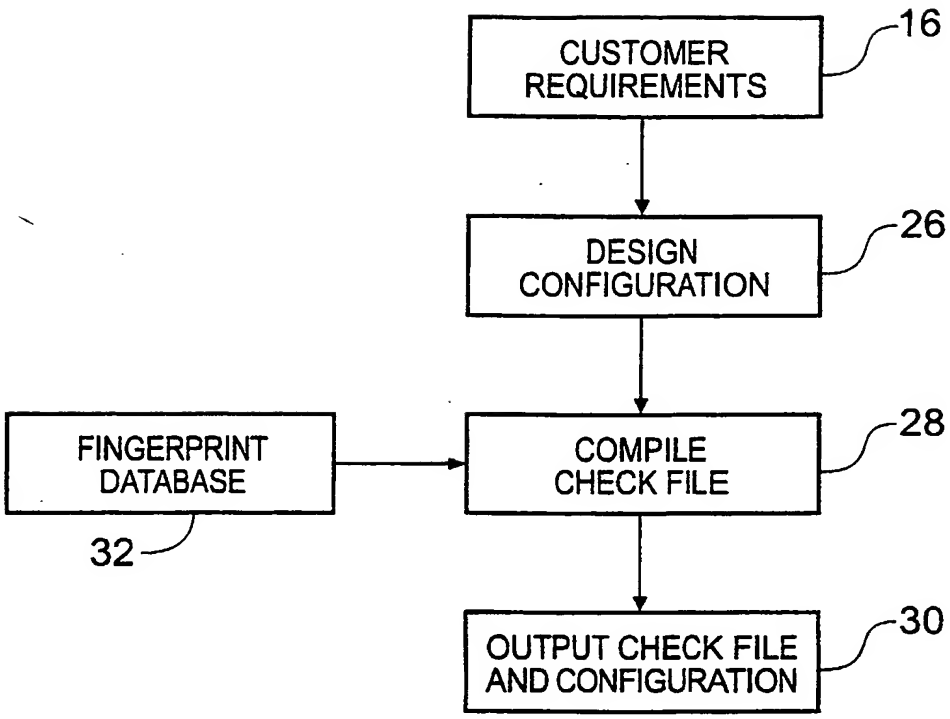
CUSTOMER REQUIREMENTS
16
DESIGN CONFIGURATION
26
FINGERPRINT DATABASE
32
COMPILE CHECK FILE
28
OUTPUT CHECK FILE AND CONFIGURATION
30

Fig. 3

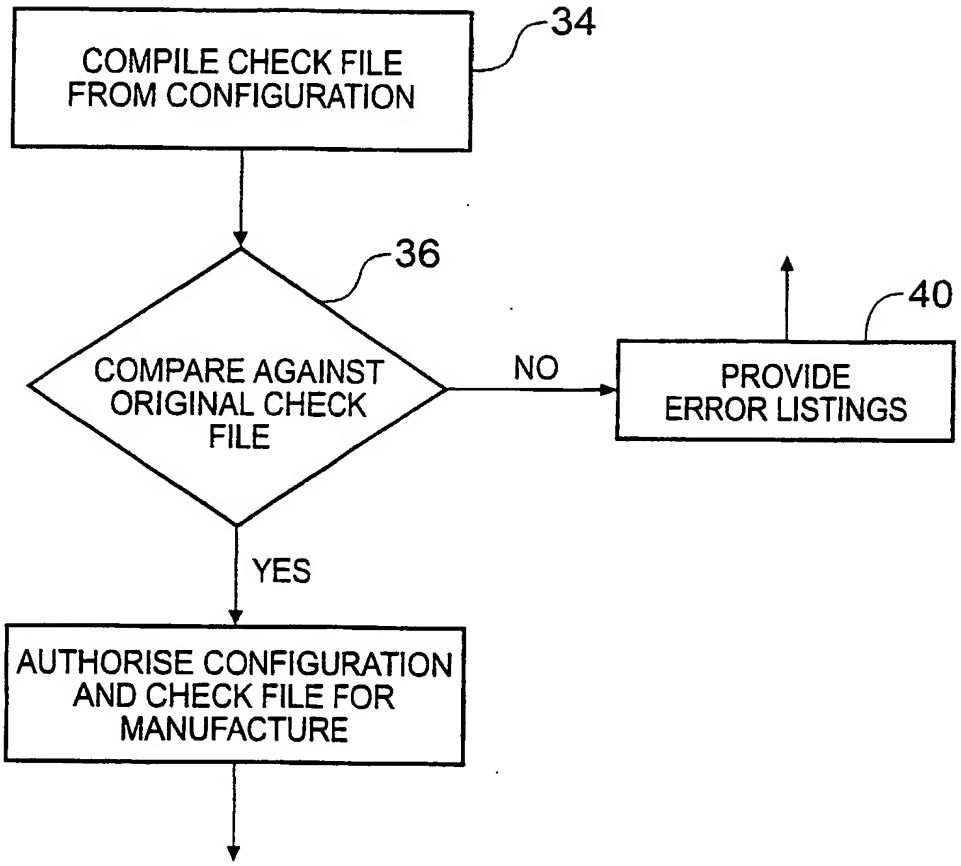
COMPILE CHECK FILE FROM CONFIGURATION
34
36
COMPARE AGAINST ORIGINAL CHECK FILE
NO
PROVIDE ERROR LISTINGS
40
YES
AUTHORISE CONFIGURATION AND CHECK FILE FOR MANUFACTURE

Fig. 4

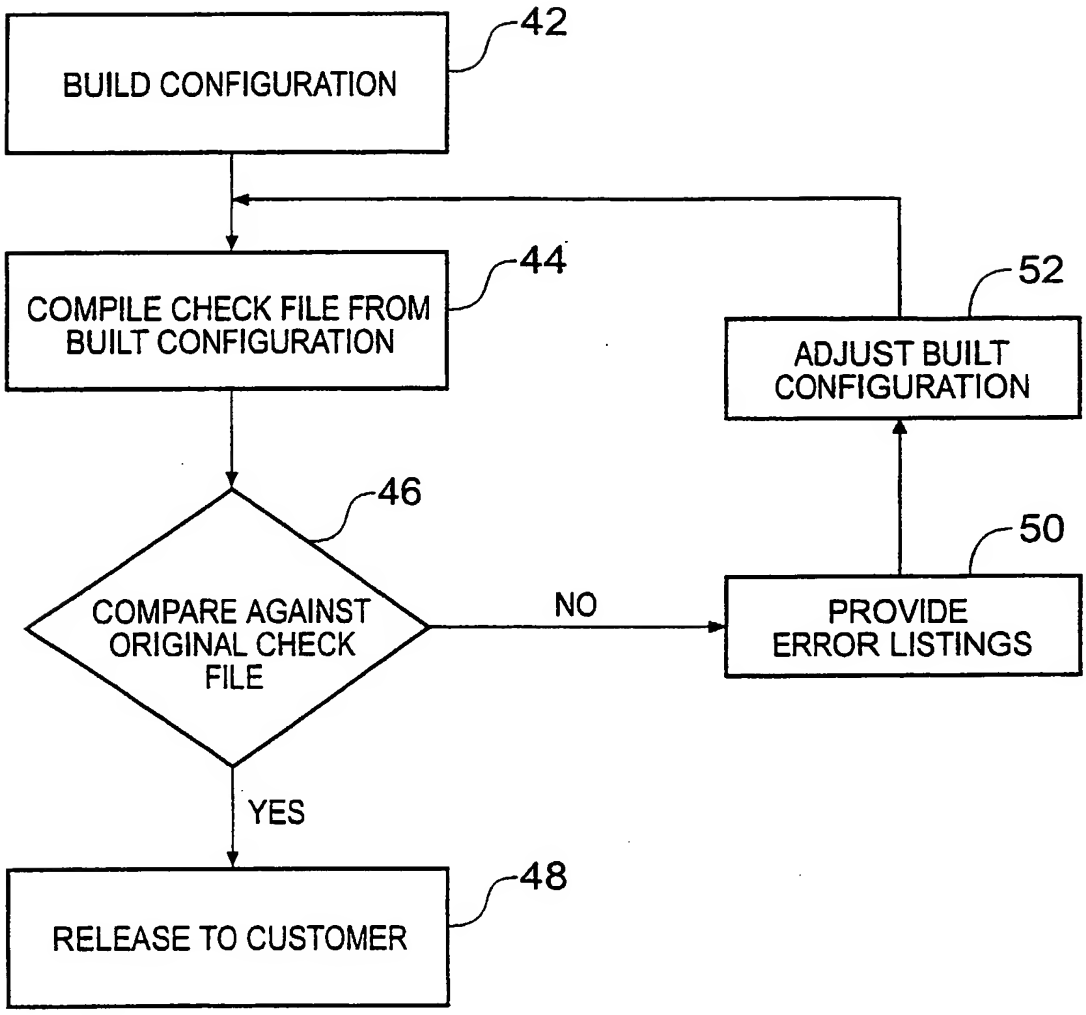
BUILD CONFIGURATION
42
COMPILE CHECK FILE FROM BUILT CONFIGURATION
44
COMPARE AGAINST ORIGINAL CHECK FILE
46
NO
YES
PROVIDE ERROR LISTINGS
50
ADJUST BUILT CONFIGURATION
52
RELEASE TO CUSTOMER
48

Fig. 5

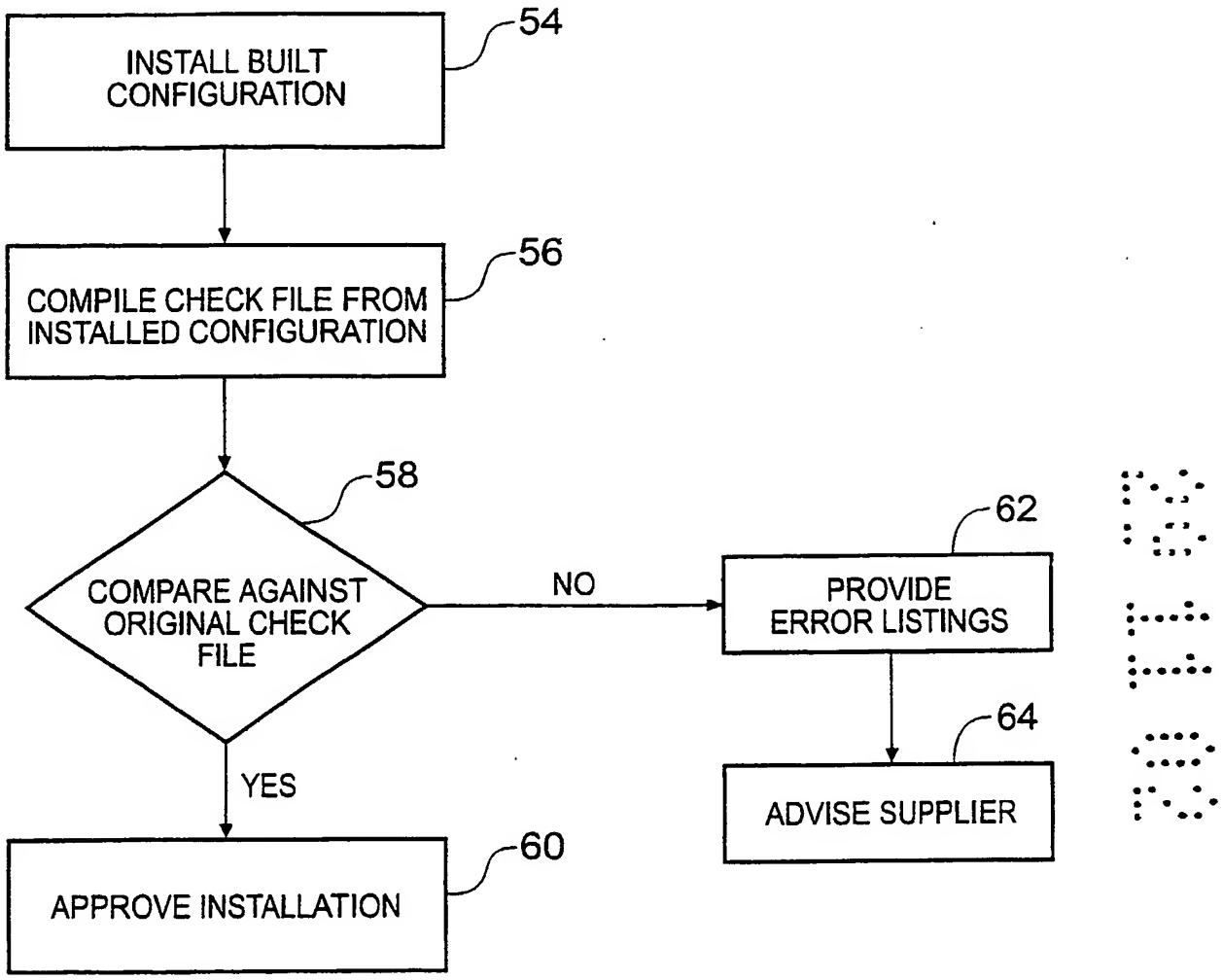
INSTALL BUILT CONFIGURATION
54
COMPILE CHECK FILE FROM INSTALLED CONFIGURATION
56
COMPARE AGAINST ORIGINAL CHECK FILE
58
NO
PROVIDE ERROR LISTINGS
62
ADVISE SUPPLIER
64
YES
APPROVE INSTALLATION
60

Fig. 6

METHOD FOR CHECKING A COMPUTER SYSTEM CONFIGURATION

Field Of The Invention

This invention relates to a method for checking a computer system configuration, and in particular for verifying either a software configuration, or a firmware configuration, or a hardware configuration, or any combination of these.

Background to the Invention

Computer systems of all types have increased greatly in complexity in recent years. One result of this increase in complexity is the tailoring of hardware, software and firmware configurations of computers to particular end users. This now happens with all types of computer including PCs and servers.

When a user orders a computer a particular configuration is specified. This will comprise a hardware configuration, firmware in which reside many of the basic routines required to make the hardware run, and a software configuration or a software stack which includes all the software the user specifies including the particular operating system he wishes to use, any optional modules to that operating system to give particular types of enhanced functionality and usually a number of application programmes which run within the operating system.

The hardware configuration comprises a number of components, some of which have optional settings which are set at the factory when the computer is manufactured to conform with a particular user's requirement.

The firmware comprises a number of components, mostly memory units of the read only type pre-loaded with software to run various routines operations on the hardware configuration, in dependence on the hardware configuration selected. The software configuration usually includes an operating system for the hardware and a number of application programmes. These are contained as a software stack which usually resides on a hard disc in the computer system. These programmes are stored in files on the hard disc or other storage medium.

At present, there are no tools available for checking at the point of manufacture whether or not a hardware, firmware, or software configuration has been correctly installed to fulfil a customer's requirements. With the fast rate of development of computer technology this can lead to installation errors in the configurations for example incompatibility between a software file and the latest version of a particular version of hardware. If these are not picked up at manufacture, they do not become apparent until a computer system is installed at its end destination and use of it is commenced. Even then, some parts of the configuration are not accessed particularly frequently and it therefore may be some time before the error is detected.

Summary of the Invention

Data for checking that a hardware configuration is correctly installed usually comprises a data file or set of data files including data identifying the particular pieces of hardware which should be installed in a configuration and any factory settings which should be applied to the hardware.

Software and firmware can include relatively large files storing routines and applications to be executed on the computer system. Providing a corresponding file against which to check a particular configuration would require a large amount of storage and, furthermore, a significant amount of time to compare all the files installed in the configuration against a baseline configuration.

Accordingly, Sun Microsystems has developed a tool known as a fingerprint database which is publically available and contains checking data including checksums for all files available in Sun packages. A checksum is a value derived using digital logic from the data stored in a file. Checksums have been well known for many years as a way of verifying digital data. The size of a checksum need bear no relation to the size of the data file from which it is derived. In their simplest forms, checksums are single bits added to the ends of data words. In more sophisticated forms, they will comprise a plurality of bits or words added to data. The use of checksums to verify the integrity of a data file requires relatively little storage compared to the size of the file. What is required is a microprocessor or some form of digital logic to perform the checksum calculation on the file being checked to determine a checksum or comparison with a known checksum for that file. Many computer systems are capable of running this type of checking operation on their own configurations.

To date, the fingerprint database including all the checksums for Sun Microsystems Inc has not been used in manufacture.

In a specific embodiment of the present invention, a check file is derived for each computer system configuration to be produced. The check file is then used to check the configuration of the computer system to

which it relates at the point of manufacture. The check file can also be supplied to a customer and when a computer system is installed at a customer's site, the check file can be compared against a check file derived from the computer system configuration to determine whether or not there are any inaccuracies in the configuration.

The check file can be used to check hardware configuration, firmware configuration, or software configuration or any combination of these.

The check file is originally generated when the computer system configuration is defined. It is derived directly from the configuration. If a fingerprint database is available containing checksums for all files available, this can be accessed to construct the check file. If such a database is not available then checksums from data files can be derived using digital logic, and predetermined checksum routines.

At manufacture, embodiments of this invention enable a variety of computer system configurations to be checked simultaneously whilst they are still in the factory. This can be the checking of a plurality of identical configurations or of different configurations. A copy of the check file for each configuration can also be supplied with each computer system delivered to each end user and a further verification of the integrity of the configuration performed either by the end user or by an installation engineer before the system comes into operation.

Computer systems can be arranged to check their own configurations using the check file, or, alternatively, a further computer system can be used to compare the configuration of the computer system against the check file.

Brief Description of the Drawings

Specific embodiments of the invention will now be described in detail by way of example with reference to the accompanying drawings in which :

Figure 1 shows schematically the relationships between hardware, firmware, and software in a computer configuration;

Figure 2 shows schematically the production of a particular configuration from the customer requirements stage through to a customer installation being in place;

Figure 3 shows how at the engineering stage a check file is produced in an embodiment of the invention;

Figure 4 shows the quality assurance processes applied to the check file and configuration generated in figure 3 in an embodiment of the invention;

Figure 5 shows the manufacturing checking processes applied using the check file generated in figure 3; and

Figure 6 shows the checking processes performed at a customer installation.

Detailed Description of Preferred Embodiments

In figure 1, a computer configuration is shown. This comprises a hardware configuration 10, a firmware configuration 12 and a software stack 14. Each of these three portions of the configuration are in communication with each other. As described above, the hardware configuration comprises all the hardware elements required to meet a particular customer requirement, the firmware configuration comprises the routines required to make the particular hardware configuration run, and is normally stored in some form of read-only memory. The software stack comprises a set of files which will typically include an operating system and a set of

application programmes which are required by the customer to run on the hardware.

Figure 2 illustrates the flow from a customer's requirements being specified at 16 through to the customer installation at 18. Customer requirements are firstly provided to an engineering department at a manufacturer which designs a hardware, firmware, and software configuration at 20 to meet the particular customer requirements. At the same time, a check file is generated which includes listings of all the hardware items considered appropriate, the firmware requirements, including pre-loaded routines, and data derived from the files comprising the software stack 14. The hardware listings include part numbers and details of optional settings to the hardware. The firmware listing also include part numbers, but also include checksum data derived from routines stored in firmware. The data derived from the software stack includes checksum data. This is data derived logically from the bits comprising the file using known techniques. The checksum data for a file will require a much smaller amount of storage than the file itself. Checksum data can also be derived from the routine stored in firmware.

At Sun Microsystems, a fingerprint database is available which stores checksum files for every application programme or software routine produced by Sun Microsystems. Usually, the engineering department will access this database to compile the check file using the checksums for each file in the software stack or routine stored in firmware.

The process then passes to a quality assurance department 22 which compiles a check file from the configuration supplied. It does this by checking the hardware present and any particular settings of this hardware, checking the firmware installed and deriving

checksum data from routine stored in firmware, and deriving checksum data from the files in the software stack. The checksum data is derived from first principles using the same routines as should have been used at the engineering stage 20.

Once the configuration has been approved by the quality assurance department 22 it passes to the manufacturing department 24 which builds the particular configuration. After building the configuration it also generates a check file from first principles using the same checksum logic as used by the engineering department and quality assurance department. This check file is compared against the original check file and if it corresponds the computer is released to a customer.

At a customer installation 18, a check file is generated again from first principles using the same checksum logic as used in the earlier stages and if the check file generated corresponds to that from the earlier stages, the installation is approved.

The process shown at 20 in figure 2 is illustrated in more detail in figure 3. In this, a customer's requirement 16 are received and a configuration to meet these requirements is designed at 26. A check file is then compiled at 28. This is derived from the hardware specified by the configuration and its settings, the firmware specified and the software stack specified. Routines stored in firmware will have checksum data in the check file and each file in the software stack will also have checksum data for it stored in the check file. The checksum data in an organisation such as Sun Microsystems is read directly to the check file from a fingerprint database which stores checksum data for all files available within Sun Microsystems. Once the check file has been compiled at 28, it and the configuration

are output at 30 and can be supplied to the quality assurance department 22.

At the quality assurance department, a new check file is derived from the configuration specified. This check file comprises the data relating to the hardware configuration and settings applied thereto, data relating to the firmware, including checksum data for routines stored therein, and checksum data derived from the software stack. Checksum data is derived from first principles from the configuration using the same checksum routines as were originally used to derive the checksum data stored in the fingerprint database 32. This takes place at step 34.

Once the new check file has been compiled, it is compared against the original check file compiled by the engineering department at 36. If the check files passes the comparison test then the configuration and check file are authorised for manufacture at 38 and are released to the manufacturing department.

If the comparison test is not passed, a listing of errors is provided at 40 and these are returned to the engineering department at 20 to check the configuration against that specified and to ensure that the errors detected are corrected either by respecifying the configuration or by correcting the check file.

The manufacturing process is shown in figure 5, and in this, the configuration specified is built at 42. At [illegible] a check file is derived from the built configuration [illegible]ing first principles by examining the hardware in the configuration and the settings applied thereto, examining the firmware and the routines stored therein, and by examining the files stored in the software stack. Checksum data is derived from the routines stored in firmware and from the files in the software stack.

The check file is compiled using an external computer which runs the routines required to generate the check file from the configuration that has been built. Alternatively the check file can be generated directly by the computer system that has been built using a software diagnostic that runs the routines to generate the check file.

The newly derived check file is then compared against the original check file provided by the engineering department at 46. If for a particular built machine the comparison is passed then the machine and the check file are released to a customer at 48. If the comparison is failed then error listings are provided at 50 and the built configuration is checked and adjusted at 52. Flow then passes again to the compilation of a check file from the adjusted configuration at 44 and further comparison against the original check file at 46.

At a customer site, the built configuration is installed 54. A check file is then generated from the installed configuration at 56 using the same methods as used at manufacture. This is done using software provided with the check file which includes software to run the checksum logic required to derive the checksum data stored for the firmware routines and for the files in the software stack. This can be provided as an internal diagnostic tool or can be run from an external machine.

A newly compiled check file is then compared and run against the original check file at 58. If the comparison is passed then the installation is approved at 60. If it is failed, then error listings are provided at 62 and the supplier advised at 64 so that remedial action can be taken for the particular configuration.

The check file described above, can be used only to check hardware, or only to check firmware, or only to

check the software stack, at any stage in the process. Alternatively, any combination of these three could be checked at any stage.

The checksum data used with the software stack is particularly important since software is frequently updated. Thus, the software installed at the manufacturing stage may not be the same software as was originally specified by the engineering department and on which the configuration is based.

The checksum data for each file will include:

file name
checksum
file size
time of last modification
file type.

Other information can also be included.

At the manufacturing stage, where a plurality of similar installations are being built, statistical analysis on errors can identify systematic problems in the manufacturing process. Thus, it will soon become apparent if software has changed since the check file being used was first generated. Changes in firmware and hardware can happen at any time and these also can be identified by statistical analysis of errors in the comparison stage. In instances where a customer is installing equipment itself, the comparisons shown in figure 6 are run and a list of hardware, firmware and software configurations produced to send back to the manufacturer for checking.

Furthermore, a customer is able to check software updates against the original check file provided with the installed system to determine exactly what parts of the installed system have been changed by updates. This is useful in analysing problems at an installation.

Although method and systems consistent with the present invention have been described with reference to the embodiment above, those skilled in the art will know of various changes in form and detail which may be made without departing from the present invention as defined in the appended claims and their full scope of equivalence.

Claims

1. A method for checking a computer system configuration comprising the steps of:

providing a first check file with a computer system configuration;

deriving a second check file from the computer system configuration;

comparing the first and second check files; and

approving the computer system configuration in dependence on the results of the comparison.

2. A method according to claim 1 in which the check file includes data relating to a hardware portion of the computer system configuration.

3. A method according to claim 1 in which the check file includes data relating to a firmware portion of the computer system configuration.

4. A method according to claim 1 in which the check file comprises data relating to a software portion of the computer system configuration.

5. A method according to claim 4 in which the step of deriving the check file comprises the step of deriving checksum data from said software portion.

6. A method according to claim 5 in which the step of deriving checksum data comprises the step of deriving checksum data for each file in the software configuration.

7. A computer program product comprising a first check file and executable computer code which when loaded on a computer cause it to execute the steps of:

deriving a second check file from a computer system configuration corresponding to the configuration from which the first check file was derived;

comparing the first and second check files; and

approving the computer system configuration in dependence on the result of the comparison.

8. A computer programme product according to claim 7 in which the first and second check files comprise checksum data derived from a software portion of the computer system configuration.

9. A computer program product according to claim 7 and 8 stored within a computer system.

10. A computer system comprising a computer with a predetermined configuration, and a computer program product according to claim 7 or 8.

11. A method for checking a specification of a computer system configuration comprising the steps of:

providing a first check file with the specification;

deriving a second check file from the specification;

comparing the first and second check files; and

approving the specification in dependence on the results of the comparison.

12. A method according to claim 11 in which the first check file includes checksum data derived from a software portion of the specification , and the step of deriving the second check file includes the step of deriving checksum data from the software portion of the specification.

13. A method of manufacturing a computer system configuration including the steps of checking a manufactured

computer system configuration according to the method of any of the claims 1 to 6.

14. A method for checking a computer system configuration after installation for a customer comprising the steps of:

providing a first check file with the computer system configuration;

deriving a second check file from the computer system configuration after installation;

comparing the first and second check files; and

approving the installation in dependence on the results of the comparison.

15. A method according to claim 14 in which the first and second check files each include checksum data.

Amendments to the claims have been filed as follows

15

1. A method for checking a computer system configuration comprising the steps of:

specifying a set of customer's requirements for a computer system configuration;

designing a computer system configuration meeting the specified requirements;

generating a first check file relating to the designed computer system configuration;

building the designed computer system configuration;

deriving a second check file from the built computer system configuration;

comparing the first and second check files; and

approving the built computer system configuration in dependence on the results of the comparison.

2. A method according to claim 1 in which the first check file includes data relating to a hardware portion of the designed computer system configuration.

3. A method according to claim 1 in which the first check file includes data relating to a firmware portion of the designed computer system configuration.

4. A method according to claim 1 in which the first check file comprises data relating to a software portion of the designed computer system configuration.

5. A method according to claim 4 in which the step of generating the first check file comprises the step of generating checksum data from said software portion.

6. A method according to claim 5 in which the step of generating checksum data comprises the step of deriving checksum data for each file in said software portion.

7. A method according to claim 1 in which the second check file includes data relating to a hardware portion of the built computer system configuration.

8. A method according to claim 1 in which the second check file includes data relating to a firmware portion of the built computer system configuration.

9. A method according to claim 1 in which the second check file comprises data relating to a software portion of the built computer system configuration.

10. A method according to claim 9 in which the step of generating the second check file comprises the step of generating checksum data from said software portion.

11. A method according to claim 10 in which the step of generating checksum data comprises the step of deriving checksum data for each file in said software portion.

12. A method for checking a computer system configuration comprising the steps of:

specifying a set of customer's requirements for a computer system configuration;

designing a computer system configuration meeting the specified requirements;

generating a first check file relating to the designed computer system configuration;

building and installing the designed computer system configuration;

deriving a second check file from the installed computer system configuration;

comparing the first and second check files; and

approving the installed computer system configuration in dependence on the results of the comparison.

13. A method according to claim 12 in which the first check file includes data relating to a hardware portion of the designed computer system configuration.

14. A method according to claim 12 in which the first check file includes data relating to a firmware portion of the designed computer system configuration.

15. A method according to claim 12 in which the first check file comprises data relating to a software portion of the designed computer system configuration.

16. A method according to claim 15 in which the step of generating the first check file comprises the step of generating checksum data from said software portion.

17. A method according to claim 16 in which the step of generating checksum data comprises the step of deriving checksum data for each file in said software portion.

18. A method according to claim 12 in which the second check file includes data relating to a hardware portion of the installed computer system configuration.

19. A method according to claim 12 in which the second check file includes data relating to a firmware portion of the installed computer system configuration.

20. A method according to claim 12 in which the second check file comprises data relating to a software portion of the installed computer system configuration.

21. A method according to claim 20 in which the step of generating the second check file comprises the step of generating checksum data from said software portion.

22. A method according to claim 21 in which the step of generating checksum data comprises the step of deriving checksum data for each file in said software portion.

B2

INVESTOR IN PEOPLE

Application No: GB 0121586.2 **Examiner:** David P Maskery
Claims searched: 1- 22 as amended **Date of search:** 29 April 2003

Patents Act 1977 : Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance	
X, Y	X 1 - 4, 7 - 9, 12 - 15, 18 - 20 Y 5, 6, 10, 11, 16, 17, 21 & 22	GB 2353373 A	(DELL) See page 3 lines 16 - 29 and page 9 line 15 - 25.
X, Y	X 1 - 4, 7 - 9, 12 - 15, 18 - 20 Y 5, 6, 10, 11, 16, 17, 21 & 22	US 5963743	(DELL) See col 4 lines 11 - 26 and col 6 lines 41 - 54
X	1 & 12	GB 2203869 A	(APPLE) See page 13.
Y	5, 6, 10, 11, 16, 17, 21 & 22	US 6223272 B1	(SIEMENS) See whole document.

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC[V]:

G4A

Worldwide search of patent documents classified in the following areas of the IPC[7]:

G06F

The following online and other databases have been used in the preparation of this search report:

EPODOC, JAPIO, WPI.